FACTA UNIVERSITATIS (NIŠ)
SER. MATH. INFORM. Vol. 30, No 1 (2015), 13–27

A SEQUENTIAL DUAL METHOD FOR THE STRUCTURED RAMP LOSS MINIMIZATION *

Dejan Mančev

Abstract. The paper presents a sequential dual method for the non-convex structured ramp loss minimization. The method uses the concave-convex procedure which transforms a non-convex problem iteratively into a series of convex ones. The sequential minimal optimization is used to deal with the convex optimization by sequentially traversing through the data and optimizing parameters associated with the incrementally built set of active structures inside each of the training examples. The paper includes the results on two sequence labeling problems, shallow parsing and part-of-speech tagging, and also presents the results on artificial data when the method is exposed to outlayers. The comparison with a primal sub-gradient method with the structured ramp and hinge loss is also presented.

1. Introduction

Support vector machines (SVMs) [18] usually assume a convex loss during the optimization. The convexity contributes to an easier optimization which ends up in the global optimum. The binary version of a SVM is extended to the structured version [17] by optimizing the structured hinge loss and retaining its convex property. Structured classifiers allow better incorporation of features into the learning procedure since they can deal with the connections between parts the structure is made of, which results in better recognition results in comparison to standard binary and multiclass classifiers. Over the last decade many algorithms adapted to structured version have been written, such as the Perceptron algorithm [5], the passive-aggressive algorithm [7], the sequential dual method [1], the Pegasos algorithm [14], etc.

In parallel, there has also been work on non-convex losses. The non-convex ramp loss is defined in [6] for binary classification with the aim to avoid that examples with a higher hinge loss become support vectors. In this way, a sparser

Received December 25, 2014.; Accepted January 27, 2015.
2010 *Mathematics Subject Classification.* Primary 68T50; Secondary 90C26, 68T10
*This research was supported by Ministry of Education, Science and Technological Development, Republic of Serbia, Grant No. 174013.

model is created which has computational benefits. They use the concave-convex procedure (CCCP) [22] to optimize the non-convex loss. In order to improve the training time, the online version of SVM with the ramp loss is presented in [20], which also uses the CCCP with the strategy of an efficient working set selection. With the similar aim, [9] uses the CCCP in combination with procedures from LaSVM solver [2] to get an online solver with a significantly lower training and classification time. This area continues to be developed with other approaches and move in different directions such as introducing a smooth ramp loss [19], using linear programming for ramp loss SVMs [3], introducing heuristics to handle a mixed integer non-linear optimization for the ramp loss on larger datasets [4], etc.

In addition to a growing interest in the ramp loss in binary classification, there has been research, in parallel, for the corresponding methods in the structured case. The paper [8] introduce a structured version of the non-convex loss. In difference to the binary case, in the structured case we do not have a direct reduction of support vectors and a significant decrease in runtime. The presented advantage of the non-convex structured loss in [8] happens in the scenario in which the labels are noisy and when for each example there is a large set of labels which are (almost) as good as the label in the training set. This better performance of the structured ramp loss is due to the fact that it is upper bounded and that the noisy examples can be discarded when the error is too large, as opposite to the unbounded structured hinge loss. Next, three different types of structured ramp losses are presented in [11], which are successfully applied to problems in machine translation. The algorithm uses a combination of the CCCP with the stochastic sub-gradient descent for the parameter optimization.

In this paper, we present a sequential method for the optimization of one of the formulations of the structured ramp loss in dual space. After the application of the CCCP, the convex problem is optimized in dual space by sequentially traversing through examples one at a time, in a similar fashion as the sequential dual method presented in [1]. We present experimental results on two sequence labeling problems and also check the behavior of the structured ramp loss on noisy data.

2. Preliminaries and ramp loss characteristics

Let $\mathscr{D} = ((\boldsymbol{x}^n, \boldsymbol{y}^n))_{n=1}^N$ be a training set, where each input $\boldsymbol{x}^n$ has the corresponding output structure $\boldsymbol{y}^n$. The set of all possible structures over $\boldsymbol{x}^n$ is denoted by $\mathscr{Y}(\boldsymbol{x}^n)$. In case of sequence labeling, for example, $\boldsymbol{x}^n \in \mathcal{X}^{T_n}$ represents an input sequence of length T_n and $\mathscr{Y}(\boldsymbol{x}^n) = \mathcal{Y}^{T_n}$, where $\mathcal{Y}$ represents a set of possible labels for an element of the input alphabet $\mathcal{X}$. The problem of minimizing the regularized empirical risk over the set $\mathscr{D}$ is

$$\min_{\mathbf{w}} J(\mathbf{w}) = \min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{N}\sum_{n=1}^{N} \ell_n(\mathbf{w}), \tag{2.1}$$

where $\ell_n(\mathbf{w})$ represents a loss function on the nth example with parameters $\mathbf{w}$. In the case of the ramp loss, $\ell_n(\mathbf{w})$ is defined as

$$\begin{aligned}(2.2)\qquad \ell_n^{\text{Ramp}}(\mathbf{w}) &= \ell_n^{\text{MM}}(\mathbf{w}) + \ell_n^{\text{C}}(\mathbf{w}) \\ &= \max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell(\mathbf{w};(\boldsymbol{x}^n,\boldsymbol{y})) - \max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell_C(\mathbf{w};(\boldsymbol{x}^n,\boldsymbol{y})) \\ &= \ell(\mathbf{w};(\boldsymbol{x}^n,\widetilde{\boldsymbol{y}}_{n,\mathbf{w}})) - \ell_C(\mathbf{w};\left(\boldsymbol{x}^n,\overline{\boldsymbol{y}}_{n,\mathbf{w}}\right)),\end{aligned}$$

where[1]

$$\begin{aligned}\ell_C(\mathbf{w};(\boldsymbol{x}^n,\boldsymbol{y})) &= \max\{0,-L(\boldsymbol{y}^n,\boldsymbol{y})-\mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y})\},\ \ell_n^{\text{C}}(\mathbf{w}) = -\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell_C(\mathbf{w};(\boldsymbol{x}^n,\boldsymbol{y}))\,, \\ \ell(\mathbf{w};(\boldsymbol{x}^n,\boldsymbol{y})) &= \max\{0, L(\boldsymbol{y}^n,\boldsymbol{y}) - \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y})\},\ \ell_n^{\text{MM}}(\mathbf{w}) = \max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell(\mathbf{w};(\boldsymbol{x}^n,\boldsymbol{y})),\end{aligned}$$

with $\Delta\mathbf{F}_n(\boldsymbol{y}) = \mathbf{F}(\boldsymbol{x}^n,\boldsymbol{y}^n) - \mathbf{F}(\boldsymbol{x}^n,\boldsymbol{y})$. The optimal structures $\widetilde{\boldsymbol{y}}_{n,\mathbf{w}}$ and $\overline{\boldsymbol{y}}_{n,\mathbf{w}}$, in which the parameters $\mathbf{w}$ can be omitted when there is no confusion, are defined as

$$(2.3)\qquad \widetilde{\boldsymbol{y}}_n \equiv \widetilde{\boldsymbol{y}}_{n,\mathbf{w}} = \arg\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell(\mathbf{w};(\boldsymbol{x}^n,\boldsymbol{y})),\quad \overline{\boldsymbol{y}}_n \equiv \overline{\boldsymbol{y}}_{n,\mathbf{w}} = \arg\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell_C(\mathbf{w};(\boldsymbol{x}^n,\boldsymbol{y}))\,.$$

The function $L(\boldsymbol{y}^n,\boldsymbol{y})$ represents the *cost* of assigning the output $\boldsymbol{y}$ to observation $\boldsymbol{x}^n$ instead of $\boldsymbol{y}^n$, while $\mathbf{F}(\boldsymbol{x},\boldsymbol{y})$ represents a *global feature vector* measuring the compatibility of $\boldsymbol{x}$ and $\boldsymbol{y}$.

When introducing the structured ramp loss, [8] had the aim to improve the results on data where the labels are noisy, and in the case when there are a lot of structures which are as good as the original one. This relies on some important characteristics which can be written for the ramp loss. First, from the definition of the ramp loss, we can see its relation to the hinge loss $\ell_n^{\text{Ramp}}(\mathbf{w}) \le \ell_n^{\text{MM}}(\mathbf{w})$, because $\ell_n^{\text{C}}(\mathbf{w}) \le 0$. Next, the ramp loss can not be increased without the bound, as it is case with the hinge loss, because the following inequality holds

$$(2.4)\qquad 2L(\boldsymbol{y}^n,\overline{\boldsymbol{y}}_n) \le \ell_n^{\text{Ramp}}(\mathbf{w}) \le 2L(\boldsymbol{y}^n,\widetilde{\boldsymbol{y}}_n).$$

This can be easily seen by plugin structures $\overline{\boldsymbol{y}}_n$ and $\widetilde{\boldsymbol{y}}_n$ into the ramp loss definition

$$\begin{aligned}\ell_n^{\text{Ramp}}(\mathbf{w}) &= L(\boldsymbol{y}^n,\widetilde{\boldsymbol{y}}_n) - \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\widetilde{\boldsymbol{y}}_n) + \min_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \{L(\boldsymbol{y}^n,\boldsymbol{y}) + \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y})\} \\ &\le L(\boldsymbol{y}^n,\widetilde{\boldsymbol{y}}_n) - \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\widetilde{\boldsymbol{y}}_n) + L(\boldsymbol{y}^n,\widetilde{\boldsymbol{y}}_n) + \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\widetilde{\boldsymbol{y}}_n) \\ &= 2L(\boldsymbol{y}^n,\widetilde{\boldsymbol{y}}_n)\end{aligned}$$

and

$$\begin{aligned}\ell_n^{\text{Ramp}}(\mathbf{w}) &= \max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \{L(\boldsymbol{y}^n,\boldsymbol{y}) - \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y})\} + L(\boldsymbol{y}^n,\overline{\boldsymbol{y}}_n) + \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\overline{\boldsymbol{y}}_n) \\ &\ge L(\boldsymbol{y}^n,\overline{\boldsymbol{y}}_n) - \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\overline{\boldsymbol{y}}_n) + L(\boldsymbol{y}^n,\overline{\boldsymbol{y}}_n) + \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\overline{\boldsymbol{y}}_n) \\ &= 2L(\boldsymbol{y}^n,\overline{\boldsymbol{y}}_n).\end{aligned}$$

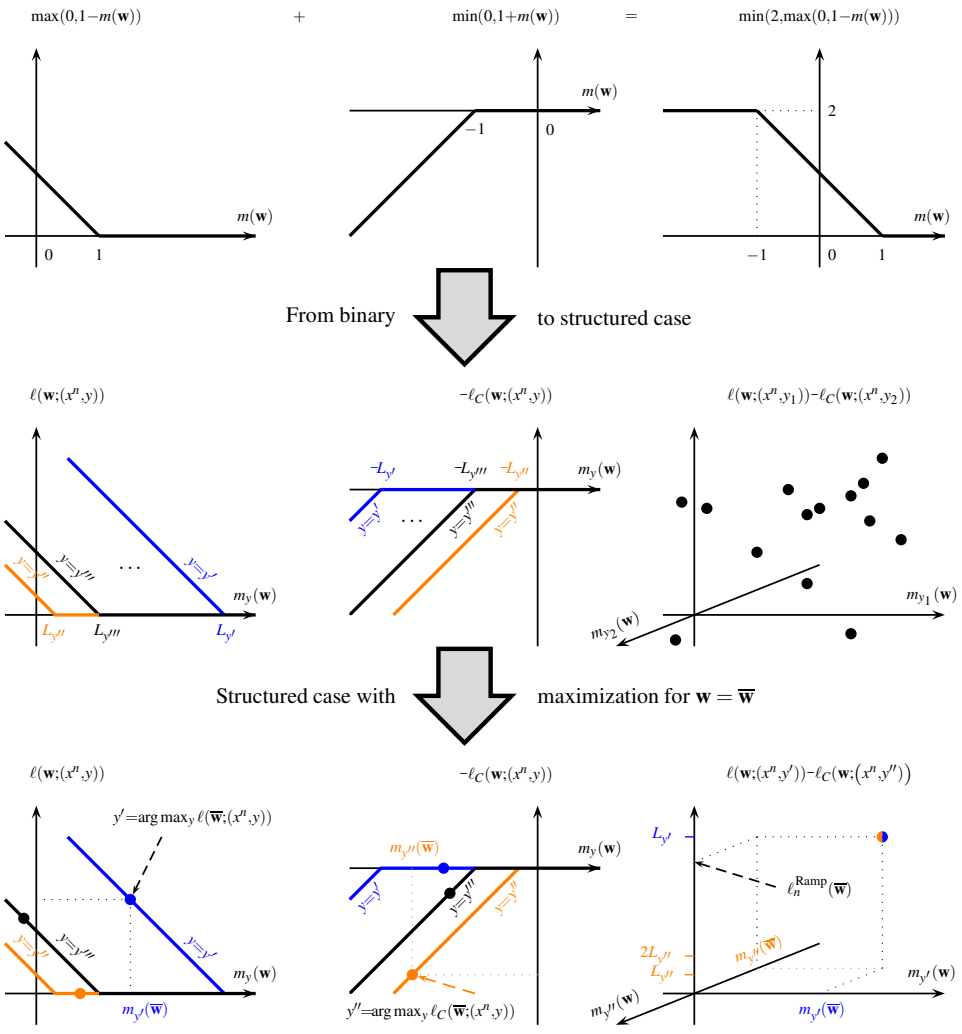

FIG. 2.1: The ramp loss in the binary and the structured case. In the binary case $m(\mathbf{w}) = t(\mathbf{w}^T\phi(x) + b)$, where $t \in \{-1, +1\}$ denotes a binary label, b represents a bias threshold, while ϕ is a nonlinear mapping from the input space to the feature space. In the structured case we denote $m_y(\mathbf{w}) = \mathbf{w}^\mathsf{T}\Delta\mathbf{F}_n(y)$ and $L_y = L(y^n, y)$. The first two panels in the middle represent the dependence of the hinge and the concave loss on the scores $m_y(\mathbf{w})$ for different structures y on the nth example, respectively, followed by the representation of the sum of these losses for the two arbitrarily chosen structures y_1 and y_2. The three bottom panels show the losses with certain parameters $\overline{\mathbf{w}}$. In that case, the ramp loss is created by the addition of the hinge and the concave loss for structures y' and y'', respectively, which are found through the maximization with parameters $\overline{\mathbf{w}}$. Then, in the bottom right panel, the ramp loss is between $2L_{y''}$ and $2L_{y'}$, i.e. the inequality $2L_{y''} \leq \ell_n^{\text{Ramp}}(\overline{\mathbf{w}}) \leq 2L_{y'}$ holds.

In the case when the cost function is bound by one, i.e., $L(\boldsymbol{y}^n, \boldsymbol{y}) \leq 1, \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)$, then we can write $0 \leq \ell_n^{\text{Ramp}}(\mathbf{w}) \leq 2$, which we have in the binary case. Figure 2.1 presents an illustration of the ramp loss in a binary and a structured case. In the binary case, when the absolute value of classifier's score $m(\mathbf{w})$ is greater than one, these examples do not become support vectors, which allows us to create a more sparse and robust model as it was done in [20]. On the other hand, in the structured case we have many structures inside one example, so for each $\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)$, we have a corresponding convex and concave function. However, in difference to the binary case, the structured ramp loss is obtained by adding convex and concave losses of different structures found during the inference with current parameters $\mathbf{w}$. So, the structured ramp loss is dependent on two structures defined with current parameters and we cannot easily apply a removal of structures similar to [20] for the binary case in order to create a sparse model. However, every point for the ramp loss will satisfy (2.4), which can help the model to become resistant to structures with noisy labels.

3. A primal-dual problem after the CCCP application on the ramp loss

With the ramp loss, we have the following optimization problem:

$$\min_{\mathbf{w}} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{N}\sum_{n=1}^{N} \ell_n^{\text{MM}}(\mathbf{w}) + \frac{C}{N}\sum_{n=1}^{N} \ell_n^{\text{C}}(\mathbf{w}) \right\}, \tag{3.1}$$

which we can optimize using the *concave-convex procedure* (CCCP) [22]. The CCCP is an iterative optimization procedure which allows us to find the optimum of a function which can be represented as a sum of the convex and the concave part. At each iteration, it approximates the concave function with its first order Taylor expansion at current parameters, and set the parameters for the next iteration as a solution of the sum of the convex and the approximate concave part.

Let us suppose that we have a loss function $J(\mathbf{w})$, which can be represented as a sum of the convex $J_{\text{vex}}(\mathbf{w})$ and the concave part $J_{\text{cav}}(\mathbf{w})$. Using the CCCP, we get series of following optimization problems

$$\mathbf{w}^{t+1} = \arg\min_{\mathbf{w}} J_t(\mathbf{w}, \mathbf{w}^t) = \arg\min_{\mathbf{w}} \left\{ J_{\text{vex}}(\mathbf{w}) + \mathbf{w}^{\mathsf{T}} J'_{\text{cav}}(\mathbf{w}^t) \right\}.$$

Let us define the *concave violation set* as

$$\mathcal{A}_{\mathbf{w}^t} = \{n : \ell_C(\mathbf{w}^t; \left(\boldsymbol{x}^n, \overline{\boldsymbol{y}}_{n,\mathbf{w}^t}\right)) > 0\}.$$

[1] Note that this is only one of possible definition of function $\ell_C(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y}))$ and the ramp loss, which is discussed in [10]. In the [8] version we have $\ell_C(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y})) = \max\{0, -\mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\boldsymbol{y})\}$.

In the case of the ramp loss, the previous CCCP iterations become

$$\mathbf{w}^{t+1} = \arg\min_{\mathbf{w}} \Big\{ \underbrace{\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{N}\sum_{n=1}^{N} \ell_n^{\mathrm{MM}}(\mathbf{w})}_{J_{\mathrm{vex}}(\mathbf{w})} + \mathbf{w}^{\mathsf{T}} \underbrace{\frac{C}{N}\sum_{n=1}^{N} \partial_{\mathbf{w}} \ell_n^{\mathrm{C}}(\mathbf{w}^t)}_{J'_{\mathrm{cav}}(\mathbf{w}^t)} \Big\}, \tag{3.2}$$

where $\partial_{\mathbf{w}} \ell_n^{\mathrm{C}}(\mathbf{w}^t) = \Delta\mathbf{F}_n(\overline{\boldsymbol{y}}_{n,\mathbf{w}^t}) \mathbb{1}[n \in \mathcal{A}_{\mathbf{w}^t}]$, $\mathbb{1}$ is the indicator function with values zero (one) if its argument is false (true) and $\overline{\boldsymbol{y}}_{n,\mathbf{w}^t}$ is calculated using parameters $\mathbf{w}^t$. Each minimization problem (3.2) can be optimized using primal sub-gradient methods, such as the structured Pegasos algorithm [14], or we can transform it to the constraint optimization problem

$$\begin{aligned} \mathbf{w}^{t+1} = \arg\min_{\mathbf{w}} J_t(\mathbf{w}, \mathbf{w}^t) = \arg\min_{\mathbf{w}} \ & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{N}\sum_{n=1}^{N} \xi_n + \frac{C}{N}\sum_{n \in \mathcal{A}_{\mathbf{w}^t}} \mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\overline{\boldsymbol{y}}_{n,\mathbf{w}^t}) \\ \text{s.t.} \quad & \mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\boldsymbol{y}) \geq L(\boldsymbol{y}^n, \boldsymbol{y}) - \xi_n, \quad \forall n, \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n). \end{aligned}$$

By introducing Lagrange multipliers $\lambda_{n,\boldsymbol{y}} \geq 0$ for each structure, we get the Lagrange function

$$\mathcal{L}(\boldsymbol{\lambda}, \mathbf{w}, \mathbf{w}^t) = J_t(\mathbf{w}, \mathbf{w}^t) - \sum_{n=1}^{N} \sum_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)} \lambda_{n,\boldsymbol{y}} \left(\mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\boldsymbol{y}) - L(\boldsymbol{y}^n, \boldsymbol{y}) + \xi_n \right).$$

From the KKT conditions [12], we get that at optimum the following must be satisfied

$$\begin{gathered} \mathbf{w} = \mathbf{u} - \mathbf{v}, \quad \mathbf{u} = \sum_{n=1}^{N} \sum_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)} \lambda_{n,\boldsymbol{y}} \Delta\mathbf{F}_n(\boldsymbol{y}), \quad \mathbf{v} = \frac{C}{N} \sum_{n \in \mathcal{A}_{\mathbf{w}^t}} \Delta\mathbf{F}_n(\overline{\boldsymbol{y}}_{n,\mathbf{w}^t}); \\ \sum_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)} \lambda_{n,\boldsymbol{y}} = \frac{C}{N}, \ \forall n; \quad \lambda_{n,\boldsymbol{y}} \left(\mathbf{w}^{\mathsf{T}} \Delta\mathbf{F}_n(\boldsymbol{y}) - L(\boldsymbol{y}^n, \boldsymbol{y}) + \xi_n \right) = 0, \ \forall n, \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n). \end{gathered}$$

Transforming the primal optimization problem into the equivalent dual one, we get

$$\min_{\boldsymbol{\lambda}} \frac{1}{2}\boldsymbol{\lambda}^{\mathsf{T}} \boldsymbol{K} \boldsymbol{\lambda} - \boldsymbol{\lambda}^{\mathsf{T}} \mathbf{L} - \boldsymbol{\lambda}^{\mathsf{T}} \Delta\mathbf{F}^{\mathsf{T}} \mathbf{v} \tag{3.3}$$

$$\text{s.t.} \sum_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)} \lambda_{n,\boldsymbol{y}} = \frac{C}{N}, \ \forall n, \ \lambda_{n,\boldsymbol{y}} \geq 0, \forall n, \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n), \tag{3.4}$$

where $\boldsymbol{K}$ is a kernel matrix defined with elements $K_{n,\boldsymbol{y},m,\boldsymbol{y}'} = \Delta\mathbf{F}_n(\boldsymbol{y})^{\mathsf{T}} \Delta\mathbf{F}_m(\boldsymbol{y}')$ for every $\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)$, $\boldsymbol{y}' \in \mathscr{Y}(\boldsymbol{x}^m)$ and $\mathbf{L}$ is a vector with elements $L_{n,\boldsymbol{y}} = L(\boldsymbol{y}^n, \boldsymbol{y})$, where $n, m \in \{1, \ldots, N\}$.

4. A sequential dual method for the structured ramp loss minimization

According to the constraints (3.4), we can restrict the optimization to only one example. Using a similar technique as described in [15] we define $\boldsymbol{\alpha}_n$ which will represent the changes in parameters $\boldsymbol{\lambda}$ on the nth example. Thus we will have changes on the nth example described as $\lambda'_{n,\boldsymbol{y}} \leftarrow \lambda_{n,\boldsymbol{y}} + \alpha_{n,\boldsymbol{y}}$. After these changes, the new parameters should be feasible, so they must satisfy $\lambda'_{n,\boldsymbol{y}} \geq 0$, while the sum of $\alpha_{n,\boldsymbol{y}}$ should be zero. Transforming the problem (3.3)-(3.4) by dropping all terms that do not depend on $\boldsymbol{\alpha}_n$, we get the optimization restricted to the nth example

$$\min_{\boldsymbol{\alpha}_n} D_{\lambda_n}(\boldsymbol{\alpha}_n) = \min_{\boldsymbol{\alpha}_n} \frac{1}{2}\boldsymbol{\alpha}_n^{\mathsf{T}}\mathbf{K}_n\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_n^{\mathsf{T}}(\mathbf{L}_n - \Delta\mathbf{F}_n^{\mathsf{T}}\mathbf{w}) \tag{4.1}$$

$$\text{s.t.} \sum_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \alpha_{n,\boldsymbol{y}} = 0, \quad \lambda_{n,\boldsymbol{y}} + \alpha_{n,\boldsymbol{y}} \geq 0, \ \forall \boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n), \tag{4.2}$$

where $\Delta\mathbf{F}_n = [\Delta\mathbf{F}_n(\boldsymbol{y})]_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)}$, and $\boldsymbol{K}_n = \Delta\mathbf{F}_n^{\mathsf{T}}\Delta\mathbf{F}_n$ is a kernel matrix for the nth example. The parameter updates on the nth example can be represented as

$$\mathbf{w}' = \mathbf{w} + \Delta\mathbf{F}_n\boldsymbol{\alpha}_n.$$

The gradient of the dual function $D_{\lambda_n}(\boldsymbol{\alpha}_n)$ with respect to $\boldsymbol{\alpha}_n$ is

$$\boldsymbol{g}_n = \mathbf{K}_n\boldsymbol{\alpha}_n - \mathbf{L}_n + \Delta\mathbf{F}_n^{\mathsf{T}}\mathbf{w} \tag{4.3}$$

with elements $g_{n,\boldsymbol{y}}$, which is used to express the violation of Karush-Kuhn-Tucker (KKT) conditions [12] for the problem (4.1)-(4.2) as

$$\max_{\boldsymbol{y}\in I_0} g_{n,\boldsymbol{y}} > \min_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} g_{n,\boldsymbol{y}}, \tag{4.4}$$

where $I_0 = \{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n) : \alpha_{n,\boldsymbol{y}} > -\lambda_{n,\boldsymbol{y}}\}$. Since all elements of $\boldsymbol{\alpha}_n$ outside the working set are equal to zero, i.e. $\alpha_{n,\boldsymbol{y}} = 0$, $\forall \boldsymbol{y} \in S_n$, the elements of the gradient can be represented in the following form

$$g_{n,\boldsymbol{y}} = \sum_{z\in S_n} \alpha_{n,z}K_{n,\boldsymbol{y},z} - L(\boldsymbol{y}^n, \boldsymbol{y}) + \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y}), \forall \boldsymbol{y} \in S_n. \tag{4.5}$$

Building the working set First, we need to find a sequence which minimizes the gradient of dual function (4.3). When we start processing the nth example, all parameters $\alpha_{n,\boldsymbol{y}}$ are equal to zero and we have that

$$\arg\min_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} -L(\boldsymbol{y}^n, \boldsymbol{y}) + \mathbf{w}^{\mathsf{T}}\Delta\mathbf{F}_n(\boldsymbol{y}) = \arg\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y})) = \widetilde{\boldsymbol{y}}_n. \tag{4.6}$$

From the previous formula, we see that structure $\widetilde{\boldsymbol{y}}_n$ can be found by applying the standard augmented Viterbi decoding using the parameters $\mathbf{w} = \mathbf{u} - \mathbf{v}$. Once the structure is found, the working set of the nth example $S_n = \{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n) \ : \ \lambda_{n,\boldsymbol{y}} > 0\}$

is incrementally built by adding the structure $\widetilde{\boldsymbol{y}}_n$ according to (4.6), in a similar fashion as described in [1]. After that, we can rewrite the KKT condition violation (4.4) with precision τ restricted to the set S_n

$$g_{n,\boldsymbol{y}''} > g_{n,\boldsymbol{y}'} + \tau, \tag{4.7}$$

$$\boldsymbol{y}' = \arg\min_{\boldsymbol{y}\in S_n} g_{n,\boldsymbol{y}}, \quad \boldsymbol{y}'' = \arg\max_{\boldsymbol{y}\in I_0'} g_{n,\boldsymbol{y}}, \tag{4.8}$$

where $I_0' = \{\boldsymbol{y} \in S_n : \alpha_{n,\boldsymbol{y}} > -\lambda_{n,\boldsymbol{y}}\}$. Note that when we start processing the nth example, $\boldsymbol{y}'$ would be equal to $\widetilde{\boldsymbol{y}}_n$, which will not be true for further iterations on the nth example, since we extend the working set S_n only when we start processing the nth example. The parameters inside each set S_n will be optimized using the sequential minimal optimization [13], where the step size is derived further.

Step size Let $\boldsymbol{h}$ be a vector where $h_{\boldsymbol{y}'} = 1$, $h_{\boldsymbol{y}''} = -1$ and all other elements are equal to zero. We can write

$$D_{\lambda_n}(\boldsymbol{\alpha}_n + \tilde{\delta}\boldsymbol{h}) = D_{\lambda_n}(\alpha_n) + \boldsymbol{h}^\mathsf{T}\boldsymbol{g}_n\tilde{\delta} + \frac{\tilde{\delta}^2}{2}\boldsymbol{h}^\mathsf{T}\nabla\boldsymbol{g}_n\boldsymbol{h}$$

which is reduced to

$$D_{\lambda_n}(\boldsymbol{\alpha}_n + \tilde{\delta}\boldsymbol{h}) = D_{\lambda_n}(\boldsymbol{\alpha}_n) + \tilde{\delta}(g_{n,\boldsymbol{y}'} - g_{n,\boldsymbol{y}''}) + \frac{\tilde{\delta}^2}{2}\|\Delta\mathbf{F}_n(\boldsymbol{y}') - \Delta\mathbf{F}_n(\boldsymbol{y}'')\|^2$$

because of the definition of $\boldsymbol{h}$. We seek $\tilde{\delta}$ which minimizes $D_{\lambda_n}(\boldsymbol{\alpha}_n + \tilde{\delta}\boldsymbol{h})$ and thus we set

$$0 = \frac{\partial}{\partial\tilde{\delta}} D_{\lambda_n}(\boldsymbol{\alpha}_n + \tilde{\delta}\boldsymbol{h}) = g_{n,\boldsymbol{y}'} - g_{n,\boldsymbol{y}''} + \tilde{\delta}\|\Delta\mathbf{F}_n(\boldsymbol{y}') - \Delta\mathbf{F}_n(\boldsymbol{y}'')\|^2$$

and get the unbounded step as

$$\tilde{\delta} = \frac{g_{n,\boldsymbol{y}''} - g_{n,\boldsymbol{y}'}}{\|\Delta\mathbf{F}_n(\boldsymbol{y}') - \Delta\mathbf{F}_n(\boldsymbol{y}'')\|^2}.$$

Now, we want $\tilde{\delta}$ to be bounded to δ in order for the change in parameters to be feasible. At that time all parameters are feasible, and we want to make a change

$$\alpha^{new}_{n,\boldsymbol{y}'} = \alpha_{n,\boldsymbol{y}'} + \delta, \qquad \alpha^{new}_{n,\boldsymbol{y}''} = \alpha_{n,\boldsymbol{y}''} - \delta \tag{4.9}$$

that must satisfy that new $\boldsymbol{\lambda}_n$ is also feasible, which means that

$$\lambda_{n,\boldsymbol{y}} + \alpha^{new}_{n,\boldsymbol{y}} \geq 0, \quad \forall\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n). \tag{4.10}$$

From (4.9) and (4.10), we get a step which must satisfy the following conditions

$$\lambda_{n,\boldsymbol{y}'} + \alpha_{n,\boldsymbol{y}'} + \delta \geq 0, \quad \lambda_{n,\boldsymbol{y}''} + \alpha_{n,\boldsymbol{y}''} - \delta \geq 0,$$

which gets us the upper and lower limit for the step size

$$-\lambda_{n,\boldsymbol{y}'} - \alpha_{n,\boldsymbol{y}'} \leq \delta \leq \lambda_{n,\boldsymbol{y}''} + \alpha_{n,\boldsymbol{y}''},$$

and the final step as

$$\delta = \max\left(-\lambda_{n,\boldsymbol{y}'} - \alpha_{n,\boldsymbol{y}'}, \min\left(\lambda_{n,\boldsymbol{y}''} + \alpha_{n,\boldsymbol{y}''}, \frac{g_{n,\boldsymbol{y}''} - g_{n,\boldsymbol{y}'}}{\|\Delta\mathbf{F}_n(\boldsymbol{y}') - \Delta\mathbf{F}_n(\boldsymbol{y}'')\|^2}\right)\right).$$

The pseudocode is presented in Algorithm 1. We initialize primal-dual parameters to the zero vector. The algorithm can also start with the given initial weights retrieved from the minimization of the hinge loss as it is done in [8]. Since $L(\boldsymbol{y}^n, \boldsymbol{y}) > 0$, for all $\boldsymbol{y} \neq \boldsymbol{y}^n$, and $L(\boldsymbol{y}^n, \boldsymbol{y}^n) = 0$, all structures $\overline{\boldsymbol{y}}_{n,\mathbf{0}}$ with parameters equal to zero will be equal to original structures $\boldsymbol{y}^n$ because

$$\overline{\boldsymbol{y}}_{n,\mathbf{0}} = \arg\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} \ell_C(\mathbf{0}; (\boldsymbol{x}^n, \boldsymbol{y})) = \arg\max_{\boldsymbol{y}\in\mathscr{Y}(\boldsymbol{x}^n)} -L(\boldsymbol{y}^n, \boldsymbol{y}) = \boldsymbol{y}^n$$

and, thus, the first CCCP iteration will represent the optimization of the hinge loss. In this way, during the first epoch the parameters have been initialized with the hinge loss for the next epochs. Note that this initialization is due to the definition of the ramp loss, and with a different formulation, as presented by [8], the previous will not hold and in that case the parameters must be explicitly initialized.

Algorithm 1: Sequential dual method for structured ramp loss

Input : Training data: $\mathscr{D} = \left((\boldsymbol{x}^n, \boldsymbol{y}^n)\right)_{n=1}^N$, parameter $C \in \mathbb{R}^+$
Number of epochs: P, Number of CCCP iterations: T
Output: Model parameters: $\mathbf{w}$

$\mathbf{w} \leftarrow \mathbf{0}$; $\boldsymbol{\lambda} \leftarrow \mathbf{0}$;
$S_n \leftarrow \{\boldsymbol{y}^n\}$, $\lambda_{n,\boldsymbol{y}^n} \leftarrow C/N$, $\forall n = 1, \ldots, N$;
for $p \leftarrow 1$ **to** P **do**
  $\mathcal{A} \leftarrow \emptyset$;
  for $n \leftarrow 1$ **to** N **do**
    $\overline{\boldsymbol{y}}_n \leftarrow \arg\max_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)} \ \ell_C(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y}))$;
    if $\ell_C(\mathbf{w}; \left(\boldsymbol{x}^n, \overline{\boldsymbol{y}}_n\right)) > 0$ **then**
      $\mathcal{A} \leftarrow \mathcal{A} \cup \{n\}$;
  $\mathbf{v} \leftarrow \frac{C}{N} \sum_{n \in \mathcal{A}} \Delta\mathbf{F}_n(\overline{\boldsymbol{y}}_n)$;
  $\mathbf{w} \leftarrow \mathbf{w} - \mathbf{v}$;
  for $t \leftarrow 1$ **to** T **do**
    for $n \leftarrow 1$ **to** N **do**
      $\widetilde{\boldsymbol{y}}_n \leftarrow \arg\max_{\boldsymbol{y} \in \mathscr{Y}(\boldsymbol{x}^n)} \ \ell(\mathbf{w}; (\boldsymbol{x}^n, \boldsymbol{y}))$;
      $S_n \leftarrow S_n \cup \{\widetilde{\boldsymbol{y}}_n\}$;
      SMO(n, S_n, $\mathbf{w}$, $\boldsymbol{\lambda}$);

Procedure SMO(n, S_n, $\mathbf{w}$, $\boldsymbol{\lambda}$)

Local constant: KKT tolerance τ

$\boldsymbol{\alpha} \leftarrow \mathbf{0}$;
repeat /* SMO over the set S_n */
  kkt_satisfied $\leftarrow$ false;
  $K_{n,\boldsymbol{y}',\boldsymbol{y}''} \leftarrow \Delta\mathbf{F}_n(\boldsymbol{y}')^\mathsf{T} \Delta\mathbf{F}_n(\boldsymbol{y}''), \forall \boldsymbol{y}', \boldsymbol{y}'' \in S_n$; /* kernel matrix */
  $g_{n,\boldsymbol{y}} \leftarrow \sum_{\boldsymbol{z} \in S_n} \alpha_{n,\boldsymbol{z}} K_{n,\boldsymbol{y},\boldsymbol{z}} - L(\boldsymbol{y}^n, \boldsymbol{y}) + \mathbf{w}^\mathsf{T} \Delta\mathbf{F}_n(\boldsymbol{y}), \forall \boldsymbol{y} \in S_n$; /* gradient */
  $\boldsymbol{y}' \leftarrow \arg\min_{\boldsymbol{y} \in S_n} g_{n,\boldsymbol{y}}$;
  $\boldsymbol{y}'' \leftarrow \arg\max_{\boldsymbol{y} \in S_n : \alpha_{n,\boldsymbol{y}} > -\lambda_{n,\boldsymbol{y}}} g_{n,\boldsymbol{y}}$;
  if $g_{n,\boldsymbol{y}''} > g_{n,\boldsymbol{y}'} + \tau$ **then** /* if KKT cond. not satisfied */
    $\delta_{n,\boldsymbol{y}',\boldsymbol{y}''} \leftarrow (g_{n,\boldsymbol{y}''} - g_{n,\boldsymbol{y}'}) / (K_{n,\boldsymbol{y}',\boldsymbol{y}'} - 2K_{n,\boldsymbol{y}',\boldsymbol{y}''} + K_{n,\boldsymbol{y}'',\boldsymbol{y}''})$;
    $\delta \leftarrow \max\left(-\lambda_{n,\boldsymbol{y}'} - \alpha_{n,\boldsymbol{y}'}, \min\left(\lambda_{n,\boldsymbol{y}''} + \alpha_{n,\boldsymbol{y}''}, \delta_{n,\boldsymbol{y}',\boldsymbol{y}''}\right)\right)$; /* bounded step */
    $\alpha_{n,\boldsymbol{y}'} \leftarrow \alpha_{n,\boldsymbol{y}'} + \delta$; $\alpha_{n,\boldsymbol{y}''} \leftarrow \alpha_{n,\boldsymbol{y}''} - \delta$
  else
    kkt_satisfied $\leftarrow$ true;
until not kkt_satisfied;
$\lambda_{n,\boldsymbol{y}} \leftarrow \lambda_{n,\boldsymbol{y}} + \alpha_{n,\boldsymbol{y}}, \forall \boldsymbol{y} \in S_n$; /* change dual parameters */
$\mathbf{w} \leftarrow \mathbf{w} + \sum_{\boldsymbol{y} \in S_n} \alpha_{n,\boldsymbol{y}} \Delta\mathbf{F}_n(\boldsymbol{y})$; /* change primal parameters */
$S_n \leftarrow S_n \setminus \{\boldsymbol{y} : \lambda_{n,\boldsymbol{y}} = 0\}$; /* remove inactive structures */

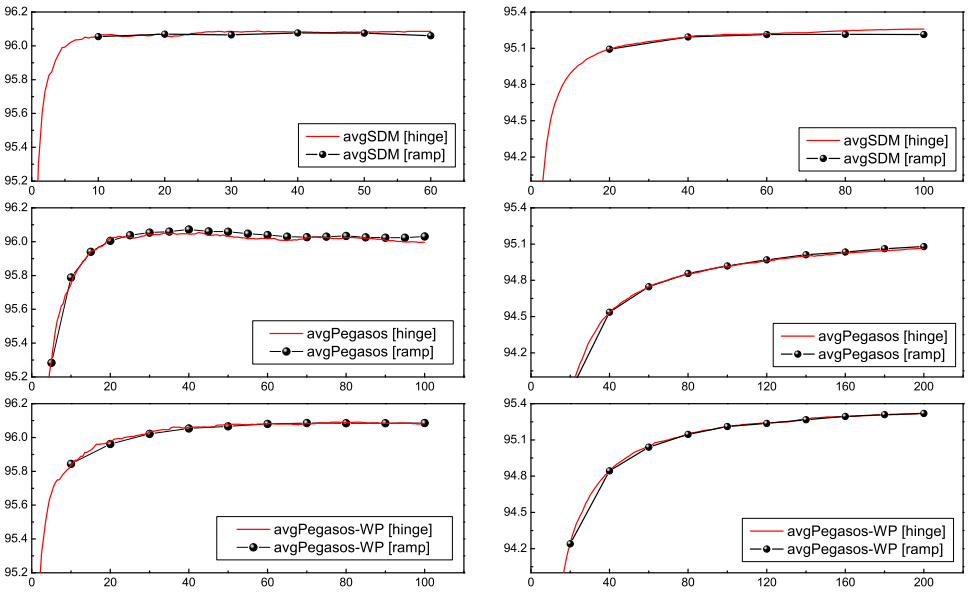

FIG. 4.1: The results for the hinge and the ramp loss for the SDM and the stochastic Pegasos algorithm. The three left panels represent the results for shallow parsing, while the three panels on the right represent the results for POS tagging. The x axis represents the number of epochs for the hinge loss, and for the ramp loss it represents the total number of epochs through data (the number of outer epochs P multiplied by the number of the inner CCCP iterations T), while the y axis represents the F-measure and accuracy for the left and right panels, respectively.

5. Experimental results

In this section, we present a comparison of the ramp loss and the hinge loss. First, we will make a comparison on two real sequence labeling problems, the shallow parsing [16] on the CONLL-2000 corpus[2] and the part-of-speech (POS) tagging on the Brown corpus[3]. After that, we will present the results obtained on artificial data, where the data is created by modifying the corpus for sequence labeling by adding outlayers in different percentage.

Compared algorithms and notation We will compare the sequential dual method (SDM) with the ramp loss and the SDM with the hinge loss [1]. In both versions, we do not use any additional heuristics to control the growth of the working set. We also include a structured Pegasos algorithm [14] for comparison. To avoid oscillations during the learning process, we have applied parameter averaging for

[2] http://www.cnts.ua.ac.be/conll2000/chunking
[3] http://khnt.aksis.uib.no/icame/manuals/brown/

Shallow parsing Method	Loss	Reg.	T	# epoch	F-measure
SDM	Hinge	$C_d = 10^{-1}$	–	100	96.084
SDM	Ramp	$C_d = 10^{-1}$	10	4	96.076
Stohastic Peg	Hinge	$\lambda = 2 \cdot 10^{-3}$	–	30	96.041
Stohastic Peg	Ramp	$\lambda = 2 \cdot 10^{-3}$	5	7	96.072
Stohastic Peg-WP	Hinge	$\lambda = 10^{-3}$	–	100	96.082
Stohastic Peg-WP	Ramp	$\lambda = 10^{-3}$	10	10	96.086
Pos tagging Method	Loss	Reg.	T	# epoch	Accuracy
SDM	Hinge	$C_d = 10^{-1}$	–	200	95.292
SDM	Ramp	$C_d = 10^{-1}$	20	4	95.216
Stohastic Peg	Hinge	$\lambda = 10^{-4}$	–	200	95.065
Stohastic Peg	Ramp	$\lambda = 10^{-4}$	20	10	95.079
Stohastic Peg-WP	Hinge	$\lambda = 2 \cdot 10^{-3}$	–	200	95.320
Stohastic Peg-WP	Ramp	$\lambda = 2 \cdot 10^{-3}$	20	10	95.319

Table 5.1: The results for the hinge vs. the ramp loss and their corresponding parameters (regularization parameters, the number of training epochs, and the number of CCCP iterations for the ramp loss) obtained from a 5-fold cross-validation for each dataset. For all algorithms, the results are presented with averaged parameters, so the *avg* prefix is omitted in the algorithm name. The parameter $C_d = C/N$ denotes the regularization parameter for SDM, and $\lambda = 1/C$ is regularization parameter used for Pegasos algorithms in [14].

all algorithms and we will denote such algorithms using the prefix *avg*. When the learning process assumes only the addition of feature vectors multiplied by an argument, parameter averaging can be easily implemented as presented by [5], while in case we need to scale the feature vector, which is the case for the Pegasos algorithm, parameters averaging can be implemented using linear transformation as described by [21]. With the suffix -WP, we will denote that the Pegasos algorithm is used without the optional projection step.

Results on real data In order to select the regularization parameter for further experiments, we perform a cross-validation. We use the 5-fold cross-validation to find the optimal parameter for each method separately, and then we employ this optimal parameter in a test scenario. Table 5.1 provides the results on both datasets with the corresponding parameters selected via cross-validation. For the hinge loss during the cross validation, we selected the best pair of the regularization parameter and the number of training epochs up to 100 (200) for shallow parsing (Pos tagging), choosing between {10, 20, 30, 50, 100} epochs (including 200 for POS tagging). During the cross-validation for the ramp loss, we selected the number of

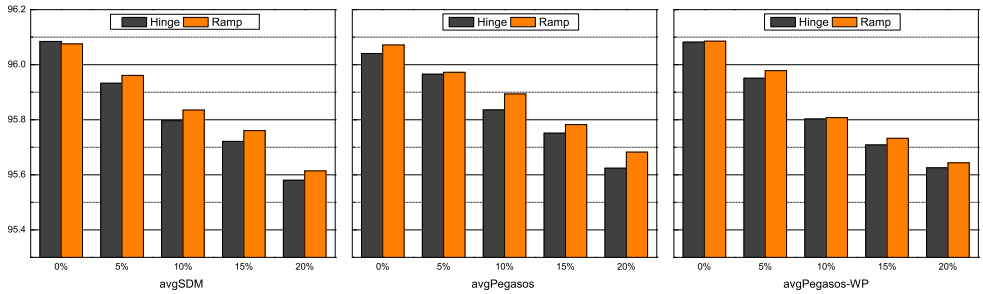

FIG. 5.1: The hinge loss vs. the ramp loss for shallow parsing on a dataset with artificially generated outlayers. The y axis represents the F-measure, while the x axis represents the percentage of examples from the CONLL-2000 corpus which are changed to outlayers. The bars corresponding to 0% on the x axis indicate the results for chunking on the corpus without modification.

CCCP iterations between {5, 10, 15, 20}. The number of epochs is selected in order that the product with the CCCP iterations be up to 100 for shallow parsing (200 for Pos tagging). In parallel, the dependence of results through epochs with optimal parameters is presented in Figure 4.1. Both from Table 5.1 and from Figure 4.1, we can notice that the results for the ramp and the hinge loss are similar for each algorithm on both datasets. The SDM reaches its highest results faster with the fewer numbers of epochs, while Pegasos without the projection step provides higher results on both datasets. The results through epochs are very close for the hinge and the ramp loss and we cannot see the advantage of the ramp loss on these corpora. Only the Pegasos algorithm without the projection performs a little bit better with the ramp loss on the shallow parsing problem.

Results on artificial data Since the ramp loss should be helpful with noisy examples, we will test the previous algorithms on this kind of data next. For this purpose, we modified the CONLL-2000 corpus by changing the labels of randomly chosen examples. We chose a portion of examples and changed their labels randomly. With this approach, such portions of examples became outlayers. Four artificial datasets were created, where randomly chosen portions of 5, 10, 15 and 20 percent of examples were affected by changing their labels. Figure 5.1 shows the results for the hinge versus the ramp loss for the SDM and the Pegasos algorithm depending on the percentage of outlayers added to corpora. We can see that all algorithms with the ramp loss perform better than the corresponding ones with the hinge loss when the learning process includes noisy examples. This is expected as the ramp loss is upper bounded, and outlayers do not make such big changes in parameters as the hinge loss can.

The results on artificial data show us that both the SDM and the Pegasos algorithm with the ramp loss clearly show an advantage over the hinge loss, which can be useful in real problems with noisy data. On the other hand, the performance

on two sequence labeling problems indicates that the algorithms perform similarly with the ramp and the hinge loss. In that case, the hinge loss has the advantage since it does not need additional decoding and it is easier for the optimization.

6. Conclusion

The paper presents a sequential dual method for the structured ramp loss. The non-convex structured ramp loss is optimized using the concave-convex procedure (CCCP). During CCCP iterations, the approximated ramp loss is optimized by sequentially traversing through data and incrementally building an active set of structures whose parameters are optimized using the sequential minimal optimization inside each example. The presented results on sequence labeling problems and the comparison with sub-gradient methods indicate that the ramp loss provides similar results to the corresponding method with the hinge loss. On the other hand, when methods are exposed to outlayers during the learning procedure, the ramp loss consistently provides better results.

REFERENCES

1. Balamurugan, P., Shevade, S., Sundararajan, S., and Keerthi, S. S. A Sequential Dual Method for Structural SVMs. In *SDM 2011 - Proceedings of the Eleventh SIAM International Conference on Data Mining* (2011).
2. Bordes, A., Ertekin, S., Weston, J., and Bottou, L. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research 6* (Dec. 2005), 1579–1619.
3. Brooks, J. P. Support vector machines with the ramp loss and the hard margin loss. *Operations research 59*, 2 (2011), 467–479.
4. Carrizosa, E., Nogales-Gmez, A., and Romero Morales, D. Heuristic approaches for support vector machines with the ramp loss. *Optimization Letters 8*, 3 (2014), 1125–1135.
5. Collins, M. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing* (2002), vol. 10, Association for Computational Linguistics, pp. 1–8.
6. Collobert, R., Sinz, F., Weston, J., and Bottou, L. Trading convexity for scalability. In *Proceedings of the 23rd International Conference on Machine Learning* (New York, NY, USA, 2006), ICML '06, ACM, pp. 201–208.
7. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. Online passive-aggressive algorithms. *The Journal of Machine Learning Research 7* (2006), 551–585.
8. Do, C. B., Le, Q. V., Teo, C. H., Chapelle, O., and Smola, A. J. Tighter bounds for structured estimation. In *Advances in neural information processing systems* (2008), pp. 281–288.
9. Ertekin, S., Bottou, L., and Giles, C. L. Nonconvex online support vector machines. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 33*, 2 (2011), 368–381.

10. Gimpel, K. *Discriminative Feature-Rich Modeling for Syntax-Based Machine Translation*. PhD thesis, Carnegie Mellon University, 2012.
11. Gimpel, K., and Smith, N. A. Structured ramp loss minimization for machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Stroudsburg, PA, USA, 2012), NAACL HLT '12, Association for Computational Linguistics, pp. 221–231.
12. Kuhn, H. W., and Tucker, A. W. Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability* (Berkeley, Calif., 1951), University of California Press, pp. 481–492.
13. Platt, J. C. Advances in kernel methods. MIT Press, Cambridge, MA, USA, 1999, ch. Fast training of support vector machines using sequential minimal optimization, pp. 185–208.
14. Shalev-Shwartz, S., Singer, Y., Srebro, N., and Cotter, A. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical Programming 127*, 1 (2011), 3–30.
15. Taskar, B. *Learning Structured Prediction Models: A Large Margin Approach*. PhD thesis, Stanford University, CA, 2004.
16. Tjong Kim Sang, E. F., and Buchholz, S. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7* (2000), Association for Computational Linguistics, pp. 127–132.
17. Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the Twenty-first International Conference on Machine Learning* (New York, NY, USA, 2004), R. Greiner and D. Schuurmans, Eds., ICML '04, ACM.
18. Vapnik, V. *Statistical learning theory*. Wiley, 1998.
19. Wang, L., Jia, H., and Li, J. Letters: Training robust support vector machine with smooth ramp loss in the primal space. *Neurocomputing 71*, 13-15 (Aug. 2008), 3020–3025.
20. Wang, Z., and Vucetic, S. Fast online training of ramp loss support vector machines. In *Proceedings - IEEE International Conference on Data Mining, ICDM* (2009), pp. 569–577.
21. Xu, W. Towards optimal one pass large scale learning with averaged stochastic gradient descent. *arXiv preprint arXiv:1107.2490* (2011).
22. Yuille, A. L., and Rangarajan, A. The concave-convex procedure. *Neural Computation 15*, 4 (Apr. 2003), 915–936.

Dejan Mančev
University of Niš
Faculty of Science and Mathematics
Department of Computer Science
ViÅegradska 33, P. O. Box 224
18000, Niš, Serbia
dejan.mancev@pmf.edu.rs